BCSE103E -
Computer Programming: Java

Digital Assignment – 4

Name: Dhruv Rajeshkumar Shah
Registration No – 21BCE0611

1. Inheritance (Circle and cylinder)

CODE

```
// JAVA DA - 4
// by Dhruv Rajeshkumar Shah
// 21BCE0611

// Circle class
class Circle {
    public double radius;

    // Constructor
    public Circle(double radius) {
        this.radius = radius;
    }

    public double area() {
        return Math.PI * radius * radius;
    }

    public double perimeter() {
        return 2 * Math.PI * radius;
    }

    public double circumference() {
        return perimeter();
    }
}

// Cylinder class extended from circle
class Cylinder extends Circle {
    public double height;

    // Constructor
    public Cylinder(double radius, double height) {
        super(radius);
        this.height = height;
    }

    public double volume() {
        return area() * height;
    }
}

public class Inheritance1 {
    public static void main(String[] args) {
        Cylinder cylinder = new Cylinder(7, 10);
        System.out.println("Cylinder area: " + cylinder.area()); // Inheriting
area() from Circle
```

```
        System.out.println("Cylinder volume: " + cylinder.volume());
    }
}
```

OUTPUT

```
dhruv@Titan /c/Dhruv/VIT/Semester-3/Java/Lab/DA-4 (main)
$  cd c:\\Dhruv\\VIT\\Semester-3\\Java\\Lab\\DA-4 ; /usr/
de\\User\\workspaceStorage\\6db8f50c252a3df6c2a8b03c961b9
Cylinder area: 153.93804002589985
Cylinder volume: 1539.3804002589986
```

2. Inheritance (Savings and loan account)

CODE

```
// JAVA DA - 4
// by Dhruv Rajeshkumar Shah
// 21BCE0611

// Account class
class Account {
    private int accNo;
    private String name;
    private double balance;
    private long phoneNo;
    private String DOB;
    private String address;

    // Constructor
    public Account(int accNo, String name, double balance, long phoneNo,
String DOB, String address) {
        this.accNo = accNo;
        this.name = name;
        this.balance = balance;
        this.phoneNo = phoneNo;
        this.DOB = DOB;
        this.address = address;
    }

    // Getters
    public int getAccNo() {
        return accNo;
    }

    public String getName() {
        return name;
    }

    public double getBalance() {
        return balance;
    }

    public long getPhoneNo() {
        return phoneNo;
    }

    public String getDOB() {
        return DOB;
    }

    public String getAddress() {
```

```
        return address;
    }

    // Setters
    public void setAccNo(int accNo) {
        this.accNo = accNo;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setBalance(double balance) {
        this.balance = balance;
    }

    public void setPhoneNo(long phoneNo) {
        this.phoneNo = phoneNo;
    }

    public void setDOB(String DOB) {
        this.DOB = DOB;
    }
}

// Savings Account class extended from Account
class SavingsAccount extends Account {
    // Constructor
    public SavingsAccount(int accNo, String name, double balance, long
phoneNo, String DOB, String address) {
        super(accNo, name, balance, phoneNo, DOB, address);
    }

    // Method to deposit money
    public void deposit(double amount) {
        setBalance(getBalance() + amount);
    }

    // Method to withdraw money
    public void withdraw(double amount) {
        setBalance(getBalance() - amount);
    }

    // Fixed deposit method
    public void fixedDeposit(double amount, int years) {
        setBalance(getBalance() + amount * years);
    }
```

```
    // Liquidate fixed deposit method
    public void liquidateFixedDeposit(double amount, int years) {
        setBalance(getBalance() - amount * years);
    }
}

// Loan Account class extended from Account
class LoanAccount extends Account {
    private int interestRate;

    // Constructor
    public LoanAccount(int accNo, String name, double balance, long phoneNo,
String DOB, String address,
            int interestRate) {
        super(accNo, name, balance, phoneNo, DOB, address);
        this.interestRate = interestRate;
    }

    // Calculate interest method
    public void calculateInterest() {
        setBalance(getBalance() + getBalance() * interestRate / 100);
    }

    // Method to pay EMI
    public void payEMI(double amount) {
        setBalance(getBalance() - amount);
    }

    // Method to top up loan
    public void topUpLoan(double amount) {
        setBalance(getBalance() + amount);
    }

    // Method to repay loan
    public void repayLoan(double amount) {
        setBalance(getBalance() - amount);
    }
}

public class Inheritance2 {
    public static void main(String[] args) {
        // Savings account
        System.out.println("Savings Account");
        SavingsAccount savingsAccount = new SavingsAccount(123456789, "Dhruv
Shah", 10000, 1234567890, "01/01/2001",
                "Mumbai");
        savingsAccount.deposit(1000);
```

```
        System.out.println("Balance after deposit: " +
savingsAccount.getBalance());
        savingsAccount.withdraw(500);
        System.out.println("Balance after withdrawal: " +
savingsAccount.getBalance());
        savingsAccount.fixedDeposit(10000, 5);
        System.out.println("Balance after fixed deposit: " +
savingsAccount.getBalance());
        savingsAccount.liquidateFixedDeposit(10000, 5);
        System.out.println("Balance after liquidating fixed deposit: " +
savingsAccount.getBalance());
        System.out.println("Savings Account Balance: " +
savingsAccount.getBalance());
        System.out.println();

        // Loan account
        System.out.println("Loan Account");
        LoanAccount loanAccount = new LoanAccount(987654321, "Dhruv Shah",
10000, 1234567890, "01/01/2001", "Mumbai",
                10);
        loanAccount.calculateInterest();
        System.out.println("Balance after interest calculation: " +
loanAccount.getBalance());
        loanAccount.payEMI(1000);
        System.out.println("Balance after paying EMI: " +
loanAccount.getBalance());
        loanAccount.topUpLoan(10000);
        System.out.println("Balance after topping up loan: " +
loanAccount.getBalance());
        loanAccount.repayLoan(10000);
        System.out.println("Loan Account Balance: " +
loanAccount.getBalance());
    }
}
```

OUTPUT

```
dhruv@Titan /c/Dhruv/VIT/Semester-3/Java/Lab/DA-4 (main)
$  /usr/bin/env C:\\Program\ Files\\Java\\jdk-11.0.11\\bin\\java.exe -c
\\6db8f50c252a3df6c2a8b03c961b96d2\\redhat.java\\jdt_ws\\DA-4_b95285c8\
Savings Account
Balance after deposit: 11000.0
Balance after withdrawal: 10500.0
Balance after fixed deposit: 60500.0
Balance after liquidating fixed deposit: 10500.0
Savings Account Balance: 10500.0

Loan Account
Balance after interest calculation: 11000.0
Balance after paying EMI: 10000.0
Balance after topping up loan: 20000.0
Loan Account Balance: 10000.0
```

3. Constructors in Inheritance (Default and parameterized)
   CODE

```java
// JAVA DA - 4
// by Dhruv Rajeshkumar Shah
// 21BCE0611

// Parent class
class Parent {
    // Default constructor
    public Parent() {
        System.out.println("Parent constructor");
    }

    // Parameterized constructor
    public Parent(int x) {
        System.out.println("Parent constructor with value " + x);
    }
}

// Child class
class Child extends Parent {
    // Default constructor
    public Child() {
        System.out.println("Child constructor");
    }

    // Parameterized constructor
    public Child(int y) {
        System.out.println("Child constructor with value " + y);
    }

    public Child(int x, int y) {
        super(x);
        System.out.println("Child constructor with value " + x + " and " + y);
    }
}

// Grandchild class
class Grandchild extends Child {
    // Default constructor
    public Grandchild() {
        System.out.println("Grandchild constructor");
    }
}

public class ConstructorsInheritance {
    public static void main(String[] args) {
        // Default constructors
```

```
        Grandchild grandchild = new Grandchild();
        System.out.println();

        // Parameterized constructors
        Child child = new Child(5);
        System.out.println();
        Child child2 = new Child(5, 10);
        System.out.println();
    }
}
```

OUTPUT

```
dhruv@Titan /c/Dhruv/VIT/Semester-3/Java/Lab/DA-4 (main)
$  cd c:\\Dhruv\\VIT\\Semester-3\\Java\\Lab\\DA-4 ; /usr/b
v\\AppData\\Roaming\\Code\\User\\workspaceStorage\\6db8f50
Inheritance
Parent constructor
Child constructor
Grandchild constructor

Parent constructor
Child constructor with value 5

Parent constructor with value 5
Child constructor with value 5 and 10
```

4. Super constructor

CODE

```
// JAVA DA - 4
// by Dhruv Rajeshkumar Shah
// 21BCE0611

// Rectangle class
class Rectangle {
    public double length;
    public double breadth;

    // Constructor
    public Rectangle(double length, double breadth) {
        this.length = length;
        this.breadth = breadth;
    }

    public double area() {
        return length * breadth;
    }

    public double perimeter() {
        return 2 * (length + breadth);
    }
}

// Cuboid class extended from rectangle
class Cuboid extends Rectangle {
    public double height;

    // Constructor
    public Cuboid(double length, double breadth, double height) {
        super(length, breadth); // Super keyword is used to call the
constructor of the parent class
        this.height = height;
    }

    public double volume() {
        return area() * height;
    }
}

public class ConstructorsInheritance2 {
    public static void main(String[] args) {
        Cuboid cuboid = new Cuboid(7, 10, 5);
        System.out.println("Cuboid area: " + cuboid.area()); // Inheriting
area() from Rectangle
        System.out.println("Cuboid volume: " + cuboid.volume());
```

```
    }
}
```

OUTPUT

```
dhruv@Titan /c/Dhruv/VIT/Semester-3/Java/Lab/DA-4 (main)
$  /usr/bin/env C:\\Program\ Files\\Java\\jdk-11.0.11\\bin\\java.exe -cp C:\\User
db8f50c252a3df6c2a8b03c961b96d2\\redhat.java\\jdt_ws\\DA-4_b95285c8\\bin Construc
Cuboid area: 70.0
Cuboid volume: 350.0
```

5. Overriding and Dynamic Dispatch

CODE

```
// JAVA DA - 4
// by Dhruv Rajeshkumar Shah
// 21BCE0611

// Parent class
class ParentOverriding {
    public void print() {
        System.out.println("Parent class");
    }
}

// Child class
class ChildOverriding extends ParentOverriding {
    // Overriding the print() method
    public void print() {
        System.out.println("Child class");
    }
}

public class Overriding {
    public static void main(String[] args) {
        ParentOverriding parent = new ParentOverriding();
        parent.print(); // Parent class

        ChildOverriding child = new ChildOverriding();
        child.print(); // Child class

        ParentOverriding parentChild = new ChildOverriding();
        parentChild.print(); // Dispatching to the child class

    }
}
```

OUTPUT

```
dhruv@Titan /c/Dhruv/VIT/Semester-3/Java/Lab/DA-4 (main)
$  cd c:\\Dhruv\\VIT\\Semester-3\\Java\\Lab\\DA-4 ; /usr/bin/e
AppData\\Roaming\\Code\\User\\workspaceStorage\\6db8f50c252a3d
Parent class
Child class
Child class
```

6. Abstract classes (Shape)

CODE

```java
// JAVA DA - 4
// by Dhruv Rajeshkumar Shah
// 21BCE0611

// Abstract class shape
abstract class Shape {
    public abstract double area();

    public abstract double perimeter();
}

// Rectangle class extended from shape
class Rectangle extends Shape {
    public double length;
    public double breadth;

    // Constructor
    public Rectangle(double length, double breadth) {
        this.length = length;
        this.breadth = breadth;
    }

    public double area() {
        return length * breadth;
    }

    public double perimeter() {
        return 2 * (length + breadth);
    }
}

// Circle class extended from shape
class Circle extends Shape {
    public double radius;
```

```
    // Constructor
    public Circle(double radius) {
        this.radius = radius;
    }

    public double area() {
        return Math.PI * radius * radius;
    }

    public double perimeter() {
        return 2 * Math.PI * radius;
    }
}

public class AbstractClass {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle(7, 10);
        System.out.println("Rectangle area: " + rectangle.area());
        System.out.println("Rectangle perimeter: " + rectangle.perimeter());

        Circle circle = new Circle(7);
        System.out.println("Circle area: " + circle.area());
        System.out.println("Circle perimeter: " + circle.perimeter());
    }
}
```

OUTPUT

```
dhruv@Titan /c/Dhruv/VIT/Semester-3/Java/Lab/DA-4 (main)
$  /usr/bin/env C:\\Program\ Files\\Java\\jdk-11.0.11\\bin\\jav
db8f50c252a3df6c2a8b03c961b96d2\\redhat.java\\jdt_ws\\DA-4_b952
Rectangle area: 70.0
Rectangle perimeter: 34.0
Circle area: 153.93804002589985
Circle perimeter: 43.982297150257104
```

7. Interfaces (Phone)

CODE

```java
// JAVA DA - 4
// by Dhruv Rajeshkumar Shah
// 21BCE0611

// Parent class
class Phone {
    public void call() {
        System.out.println("Phone call");
    }

    public void sms() {
        System.out.println("Sending SMS")
    }
}

// Interface
interface Camera {
    void click();

    void record();
}

interface MusicPlayer {
    void play();

    void stop();

    void pause();
}

// Child class
class SmartPhone extends Phone implements Camera, MusicPlayer {

    public void call() {
        System.out.println("Smart phone call");
    }

    public void sms() {
        System.out.println("Smart phone sending SMS");
    }

    public void click() {
        System.out.println("Smart phone clicking photo");
    }
```

```
    public void record() {
        System.out.println("Smart phone recording video");
    }

    public void play() {
        System.out.println("Smart phone playing music");
    }

    public void stop() {
        System.out.println("Smart phone stopping music");
    }

    public void pause() {
        System.out.println("Smart phone pausing music");
    }
}

public class Interfaces {
    public static void main(String[] args) {
        SmartPhone smartPhone = new SmartPhone();
        smartPhone.call(); // Smart phone call
        smartPhone.sms(); // Smart phone sending SMS
        smartPhone.click(); // Smart phone clicking photo
        smartPhone.record(); // Smart phone recording video
        smartPhone.play(); // Smart phone playing music
        smartPhone.stop(); // Smart phone stopping music
        smartPhone.pause(); // Smart phone pausing music
    }

}
```

OUTPUT

```
dhruv@Titan /c/Dhruv/VIT/Semester-3/Java/Lab/DA-4 (main)
$  /usr/bin/env C:\\Program\ Files\\Java\\jdk-11.0.11\\bin\
db8f50c252a3df6c2a8b03c961b96d2\\redhat.java\\jdt_ws\\DA-4_
Smart phone call
Smart phone sending SMS
Smart phone clicking photo
Smart phone recording video
Smart phone playing music
Smart phone stopping music
Smart phone pausing music
```

8. Interfaces and static methods and members

CODE

```java
// JAVA DA - 4
// by Dhruv Rajeshkumar Shah
// 21BCE0611

// Interface
interface Test {
    final static int X = 10;

    public abstract void meth1();

    public abstract void meth2();

    public static void meth3() {
        System.out.println("Static method of test");
    }
}

interface Test2 extends Test {
    void meth4();
}

class My implements Test2 {
    public void meth1() {
        System.out.println("Meth1");
    }

    public void meth2() {
        System.out.println("Meth2");
    }

    public void meth4() {
        System.out.println("Meth4");
    }
}

public class Interfaces2 {
    public static void main(String[] args) {
        My obj = new My();
        obj.meth1();
        obj.meth2();
        obj.meth4();
        Test.meth3();
        System.out.println(Test.X);
    }
}
```

OUTPUT

```
dhruv@Titan /c/Dhruv/VIT/Semester-3/Java/Lab/DA-4
Storage\\6db8f50c252a3df6c2a8b03c961b96d2\\redhat
Meth1
Meth2
Meth4
Static method of test
10
```